[19] 中华人民共和国国家知识产权局

[51] Int. Cl[7]
G06F 9/22
G06F 9/30

[12] 发明专利申请公开说明书

[21] 申请号 03103038.6

[43] 公开日 2003 年 7 月 23 日

[11] 公开号 CN 1431584A

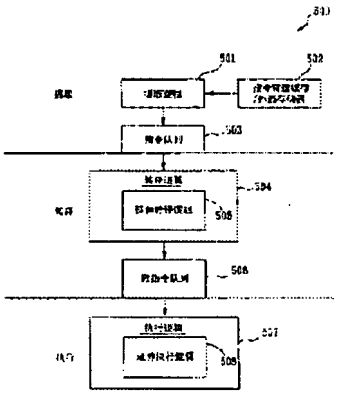
[22] 申请日 2003.1.28 [21] 申请号 03103038.6
[30] 优先权
[32] 2002. 8.22 [33] US [31] 10/227,008
[71] 申请人 智慧第一公司
地址 美国德克萨斯州
[72] 发明人 G·葛兰·亨利 罗德·E·胡克
泰瑞·派克斯
[74] 专利代理机构 隆天国际知识产权代理有限公司
代理人 楼仙英 陈 红

权利要求书 3 页 说明书 18 页 附图 9 页

[54] 发明名称 延伸微处理器数据模式的装置及方法

[57] 摘要

本发明涉及一种装置及方法，用于扩充一微处理器指令集，以提供可由该微处理器指令集的可程序化指令予以指定的延伸大小操作数。该装置包括一转译逻辑与一延伸执行逻辑。该转译逻辑将一延伸指令转译成对应的微指令，由该微处理器执行。该延伸指令具有一延伸前置码与一延伸前置码标记。该延伸前置码指定对应一指定运算的一操作数的延伸操作数大小，其中该延伸操作数大小是无法由一即有指令集进行指定。该延伸前置码标记则指出该延伸前置码，且是原本该即有指令集内另一依据架构所指定的运算码。该延伸执行逻辑耦接至转译逻辑，用以接收该对应的微指令，并使用该操作数以执行该指定运算。



ISSN1008-4274

1．一种用以延伸一微处理器的数据模式的装置，其特征在于，它包括：

一转译逻辑，用以将一延伸指令转译成对应的微指令，由微处理器执行，其中该延伸指令包括：

一延伸前置码，用以指定对应一指定运算的一操作数的延伸操作数大小，其中该延伸操作数大小不能由一既有指令集加以指定；以及

一延伸前置码标记，用以指出该延伸前置码，其中该延伸前置码标记是原本该既有指令集内另一依据架构所指定的运算码；以及

一延伸执行逻辑，耦接至该转译逻辑，用以接收该对应的微指令，并使用该操作数来执行该指定运算。

2．如权利要求1所述的装置，其特征在于所述的延伸指令还包括该既有指令集的指令项目。

3．如权利要求2所述的装置，其特征在于所述的指令项目指定该微处理器所要执行的该运算，且其中对应该运算的该操作数是提取自／储存至一缓存器，其中该缓存器依据数个操作数大小支持操作数的提取／储存。

4．如权利要求1所述的装置，其特征在于所述的延伸前置码指示该微处理器于执行该指定运算时，取代该操作数的一预设的操作数大小。

5．如权利要求1所述的装置，其特征在于所述的延伸前置码包括：

一延伸操作数大小字段，用以指定该操作数的该延伸操作数大小，其中该延伸操作数大小是该些操作数大小之一。

6．如权利要求1所述的装置，其特征在于所述的转译逻辑包括：

一逸出指令检测逻辑，用于检测该延伸前置码标记；

一指令译码逻辑，用以决定该操作数及所要执行的该运算；以及

一延伸译码逻辑，耦接至该逸出指令检测逻辑与该指令译码逻辑，用以决定该延伸操作数大小，并于该对应微指令内指定该延伸操作数大小。

7．一种扩充一既有微处理器指令集以提供延伸数据模式的机制，其特征在于，它包括：

一延伸指令，组态为指定一操作数的延伸操作数大小，该操作数对应一指定运算，而该延伸指令包括该既有微处理器指令集其中一选取的运算码，

其后则接着一 n 位的延伸前置码，该选取的运算码指出该延伸指令，而该 n 位的延伸前置码则指出该延伸操作数大小，其中该延伸操作数大小不能另依该既有微处理器指令集加以指定；以及

一转译器，组态为接收该延伸指令，并产生一微指令序列，以指示一微处理器依照该延伸操作数大小，于该操作数上执行该指定运算。

8．如权利要求 7 所述的机制，其特征在于所述的延伸指令还包括：

其余指令项目，组态为指定该操作数与该指定运算，其中该指定运算的该操作数是提供自／至一延伸操作数缓存器。

9．如权利要求 7 所述的机制，其特征在于所述的 n 位的前置码包括：

一数据模式取代字段，组态为指定该延伸操作数大小予该操作数，其中该延伸操作数大小包括数个操作数大小其中之一。

10．如权利要求 7 所述的机制，其特征在于所述的转译器包括：

一逸出指令检测器，用以检测该延伸指令内的该选取的运算码；

一指令译码器，用以译码该延伸指令的其余部分，以决定该指定运算；以及

一延伸前置码译码器，耦接至该逸出指令检测器及该指令译码器，用以译码该 n 位的延伸前置码，并于该微指令序列内指定该延伸操作数大小。

11．一种为一既有指令集增添延伸数据模式能力的指令集延伸模块，其特征在于，它包括：

一逸出标记，由一转译逻辑接收，并指出一对应指令的附随部分加以指定的一微处理器所要执行一延伸运算，其中该逸出标记为该既有指令集内的一第一运算码；

一延伸操作数大小指定元，耦接至该逸出标记，且为该附随部分其中之一，用以指定对应该延伸运算的数个数据模式其中之一；以及

一延伸执行逻辑，耦接至该转译逻辑，利用所指定的数据模式执行该延伸运算，其中该既有指令集仅提供既有的数据模式，而未能提供所指定的数据模式。

12．如权利要求 11 所述的指令集延伸模块，其特征在于所述的附随部分的其余部分包括一第二运算码与选用的数个地址指定元，用以指定该延伸运算与数个操作数，其中该些操作数是依所指定的数据模式加以执行。

13. 如权利要求 11 所述的指令集延伸模块，其特征在于所述的转译逻辑将该逸出标记与该附随部分转译成对应的微指令，该对应的微指令是指示该延伸缓存器逻辑依据所指定的数据模式，于该延伸运算执行时，存取一延伸缓存器，以提取／储存一延伸操作数。

14. 如权利要求 11 所述的指令集延伸模块，其特征在于所述的转译逻辑包括：

一逸出标记检测逻辑，用以检测该逸出标记，并指示该附随部分的转译动作需依据延伸转译常规；以及

一译码逻辑，耦接至该逸出标记检测逻辑，用以依据该既有指令集的常规，执行指令的转译动作，并依据该延伸转译常规执行该对应指令的转译，以依据所指定的数据模式，致能该延伸运算的执行。

15. 一种扩充一既有指令集架构的方法，可在一微处理器内程序化地指定一延伸数据模式，该方法包括：

提供一延伸指令，该延伸指令包括一延伸标记及一延伸前置码，其中该延伸标记是该既有指令集架构其中一第一运算码项目；

通过该延伸前置码与该延伸指令的其余部分指定该延伸数据模式与一指定运算，其中该既有指令集架构仅提供指定既有数据模式而非该延伸数据模式的指令；以及

依据该延伸数据模式执行该指定运算。

16. 如权利要求 15 所述的方法，其特征在于所述的指定延伸数据模式的动作包括：首先指定该指定运算，其中该首先指定的动作包括使用该既有指令集架构中一第二运算码项目。

17. 如权利要求 15 所述的方法，其特征在于，还包括：

将该延伸指令转译成微指令，该微指令是指示一延伸执行逻辑依据该延伸数据模式执行该延伸运算。

18. 如权利要求 17 所述的方法，其特征在于所述的转译延伸指令的动作包括：

于一转译逻辑内，检测该延伸标记；以及

依照延伸转译规则译码该延伸前置码与该延伸指令的其余部分，以取代该延伸运算的一预设数据模式。

延伸微处理器数据模式的装置及方法

与相关申请案的对照

（0001）本申请主张以下美国申请案的相关权益：案号 10 / 227008，申请日为 2002 年 8 月 22 日。

（0002）本申请与下列同在申请中的美国专利申请案有关，都具有相同的申请人与发明人。

台湾申请案号	申请日	DOCKET NUMBER	专利名称
91116957	7 / 30 / 02	CNTR：2176	延伸微处理器指令集的装置及方法
91116958	7 / 30 / 02	CNTR：2186	执行条件指令的装置及方法
91124008	10 / 18 / 02	CNTR：2187	选择性控制存储器属性的装置及方法
91116956	7 / 30 / 02	CNTR：2188	选择性地控制条件码回写的装置及方法
91116959	7 / 30 / 02	CNTR：2189	增加微处理器的缓存器数量的机制
91124006	10 / 18 / 02	CNTR：2191	延伸微处理器地址模式的装置及方法
		CNTR：2192	储存检查的禁止
		CNTR：2193	选择性中断的禁止
91124007	10 / 18 / 02	CNTR：2195	非暂存存储器参照控制机制
91116672	7 / 26 / 02	CNTR：2198	选择性地控制结果回写的装置及方法

技术领域

（0003）本发明是有关微电子的领域，尤指一种能将延伸地址模式控制纳入一既有的微处理器指令集架构的技术。

背景技术

（0004）自 1970 年代初面世以来，微处理器的使用即呈指数般成长。从最早应用于科学与技术的领域，到如今已从那些特殊领域引进商业的消费者领域，如桌上型与膝上型（laptop）计算机、视频游戏控制器以及许多其它常见的家用与商用装置等产品。

（0005）随着使用上的爆炸性成长，在技术上也历经一相对应的提升，其特征在于对下列项目有着日益升高的要求：更快的速度、更强的寻址能力、更快的存储器存取、更大的操作数，更多种一般用途类型的运算（如浮点运算、单一指令多重数据（SIMD）、条件移动等）以及附加的特殊用途运算（如数字信号处理功能及其它多媒体运算）。如此造就了该领域中惊人的技术进展，且都已应用于微处理器的设计，像扩充流水线化（extensive pipelining）、超纯量架构（super-scalar architecture）、快取结构、乱序处理（out-of-order processing）、爆发式存取（burst access）机制、分支预测（branch predication）以及假想执行（speculative execution）。总之，比起 30 年前刚出现时，现在的微处理器呈现出惊人的复杂度，且具备了强大的能力。

（0006）但与许多其它产品不同的是，有另一非常重要的因素已限制了，并持续限制着微处理器架构的演进。现今微处理器会如此复杂，一大部分得归因于这项因素，即旧有软件的兼容性。在市场考量下，所多制造商选择将新的架构特征纳入最新的微处理器设计中，但同时在这些最新的产品中，又保留了所有为确保兼容于较旧的、即所谓“旧有”（legacy）应用程序所必需的能力。

（0007）这种旧有软件兼容性的负担，没有其它地方，会比在 x86-兼容的微处理器的发展史中更加显而易见。大家都知道，现在的 32 / 16 位的虚拟模式（virtual-mode）x86 微处理器，仍可执行 1980 年代所撰写的 8 位真实模式（real-mode）的应用程序。而本领域的技术人员也承认，有不少相关的架构“包袱”堆在 x86 架构中，只为了支持与旧有应用程序及运作模式的兼容性。虽然在过去，研发者可将新开发的架构特征加入既有的指令集架构，但如今使用这

些特征所凭借的工具，即可程序化的指令，却变得相当稀少。更简单他说，在某些重要的指令集中，已没有“多余”的指令，让设计者可藉以将更新的特征纳入一既有的架构中。

（0008）例如，在 x86 指令集架构中，已经没有任何一未定义的一字节大小的运算码状态，是尚未被使用的。在主要的一字节大小的 x86 运算码图中，全部 256 个运算码状态都已被既有的指令占用了。结果是，x86 微处理器的设计者现在必须在提供新特征与保留旧有软件兼容性两者间作抉择。若要提供新的可程序化特征，则必须分派运算码状态给这些特征。若既有的指令集架构没有多余的运算码状态，则某些既存的运算码状态必须重新定义，以提供给新的特征。因此，为了提供新的特征，就得牺牲旧有软件兼容性了。

（0009）一个持续恶化且困扰微处理器设计者的问题，即是操作数的大小。早期的微处理器设计提供了使用 8 位操作数的 8 位运算。随着应用程序使用的计算日渐复杂，操作数的大小与相关的运算也增加至 16 位。现在用于桌上型／膝上型应用程序的微处理器，已能提供 32 位的操作数／运算。微处理器的操作数／运算的大小，通常称为数据模式（data mode）。因此，为了保留与旧有应用程序的兼容性，现代的桌上型／膝上型计算机的微处理器皆能以 32 位、16 位甚至是 8 位的数据模式运作。

（0010）但即使到现在，由于微处理器不能支持延伸数据模式，如 64 位与 128 位的数据模式，仍有些应用程序的领域会遭受不利的影响。不过，为了要在已无剩余运算码值的架构内支持这些延伸数据模式，必须将既有运算码重新定义，如此将会导致无法支持旧有的应用程序。